Unit Dual-Quaternion-based Pose Optimization for Visual Runway Observations

Jonghyuk Kim[1], Galen Brambley[2]

***Abstract*— This paper addresses the pose estimation problem of an aircraft runway using visual observations in a landing approach scenario. We utilize the fact that the geodetic coordinates of the most runways are known precisely with highly visible markers. Thus the runway observations can increase the level of situational awareness during the landing approach, providing additional redundancy of navigation and less reliance on global positioning system (GPS). A novel pose optimization algorithm is proposed utilizing unit dual-quaternion for the runway corner observations obtained from a monocular camera. The estimated runway pose is further fused with an inertial navigation system in an extended Kalman filter. An open-source flight simulator is used to collect and process the visual and flight dataset during the landing approach, demonstrating reliable runway pose estimates and the improved inertial navigation solution.**

I. Introduction

Pose estimation is a fundamental problem of determining the position and attitude (or orientation) of one object's coordinate system with respect to another. An extension of this principle to computer vision transforms the problem into estimating the pose between the object and the camera from which it is being observed. There has been significant progress in determining the pose of 6 degrees-of-freedom (DOF) platforms, particularly by integrating inertial measurement units (IMUs) and monocular/stereo cameras for navigation and mapping [1]–[5].

For crewed aircraft, studies have shown that the lack of positional awareness is a major cause of accidents [6]. In particular, estimating the pose of a runway relative to a monocular camera offers benefits to all aircraft which should not be underestimated. Moreover, it can be used to reduce errors in the onboard navigation system and adds a level of redundancy to the system. A considerable amount of research has gone into vision-based landing scenarios [7], [8], including the pose estimation for landings on aircraft-carrier ships [9]–[11]. Most of the work has focused on vertical-take-off-and-landing (VTOL) platforms and not much on the fixed-wing types, except the several recent works such as [12]. The reason is due to the small margin of the approaching angle in the most ship-landing scenario and thus the difficulty in obtaining precise navigational information from the cameras relative to the runway.

In the application presented here, we consider an airport runway as the object on which the aircraft is to land. Several characteristics of runways make them highly suitable as navigation aids, especially when an aircraft is on its final approach [13], [14]. First, the precise geodetic coordinates of most runways are well known, thus allowing them to be used as absolute navigation references. Second, the dimensions of runways must adhere to strict protocols, as does the white markers placed on them. Third, runways are designed to be highly visible, so in essence, they can be used as a navigation tool even if the aircraft does not intend to land at the runway it is observing. Fourth, there is no limit to the number of aircraft that can observe a given runway for navigation. The emphasis in this work is to increase the level of situational awareness for an aircraft on its final approach.

We propose a novel optimization algorithm by parametrizing the runway pose as a unit dual-quaternion. The unit dual-quaternion has been widely applied for the pose estimation problem such as strap-down inertial navigation system [15], coordination of multiple rigid-bodies [16], and pose-graph simultaneous localization and mapping (SLAM) [17]. The UDQ has the advantages of no singularity, unlike Euler angles, and fast convergence due to the unified optimization of the non-Euclidian rotation and translational vector. Extending our previous work in this area [17], we apply the UDQ framework to the runway pose estimation from monocular images. It should be noted that the reliable detection of the runway (for example, detecting the *piano-key*-like markers on the runway) is important and has been the subject of studies [18], [19], but it is not the focus in this work. The unit vectors describing the four runway corners in the camera frame are used to formulate a cost function, which is derived from the geometric relationships between the observations and runway, transformed into the dual-quaternion algebra. The pose is estimated through the Levenberg-Marquardt (LM) optimization method. The contributions of this work are

- Unit dual-quaternion parameterization to estimate the runway pose from monocular images.
- Camera-in-the-loop demonstration of visual-inertial navigation by utilizing an open-source flight simulator.

To the author's knowledge, there is no prior literature investigating the unit-dual-quaternion based optimization for the runway pose estimation problem. We also utilize a *FlightGear* simulator to demonstrate the method, as shown in Fig. 1, illustrating an example of image observations of a runway using the simulator, as well as showing the extracted visual features. The simulator is open-source and flexible, and it can be effectively used for the camera-in-the-loop

[1] Centre for Autonomous Systems, University of Technology Sydney, Australia. jonghyuk.kim@uts.edu.au

[2] Nova Systems, Canberra, Australia. galenbrambley@gmail.com

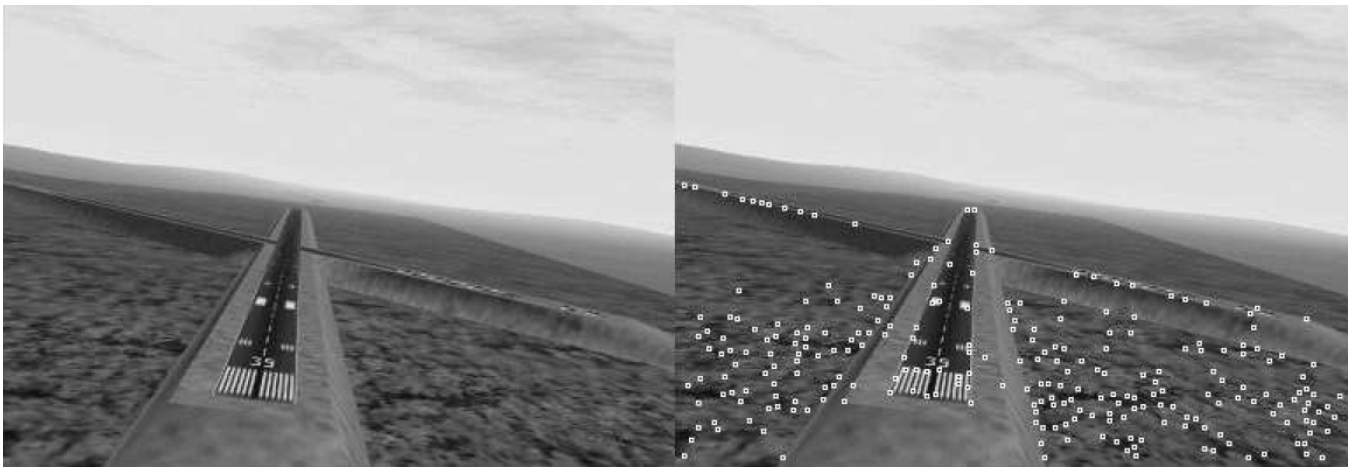

Fig. 1. An example of FlightGear visual outputs which is reproduced using a graphical user interface (GUI) window in gray scale and the detected image features (SUSAN corner features).

simulation, which we believe is a valuable tool for vision-based research.

The remainder of the article is outlined as follows. Section II provides a brief overview of the unit dual-quaternion algebra, and Section III details the unit dual-quaternion parametrization for the runway observations obtained from a monocular camera followed by the LM optimization. Section IV shows the integration of the runway observations and inertial navigation system. Experimental results are given in Section V, providing the flight simulator interface, and analyzing the optimization and filtering performance. Section VI will conclude with future direction.

II. Unit Dual-Quaternion Algebra

The unit dual-quaternion is attractive for the estimation of the pose, as the dual-quaternion can be represented as a single vector that is amenable to implementation in a statistical estimation process. In addition to this, the rotational and translational parts can be simultaneously optimized in a cost function setting, which is separable and many cases and expandable to include any number of observations. It also has a closed-form solution to the Taylor series expansion[1].

The dual-quaternion is comprised of two parts that can describe the rigid-body transformation of one coordinate frame with respect to another and is given by

$$\check{q} = q + \epsilon p \tag{1}$$

where the $\check{q}$ is used to signify the dual-quaternion, and both q and p are quaternions representing the rotation and translation, respectively, while $q = q_0 + iq_1 + jq_2 + kq_3 := [q_0,\ q_1,\ q_2,\ q_3]$[2]. The parameter ϵ is used simply to enforce a multiplicative rule onto dual-quaternions, being $\epsilon^2 = 0$. The product of two dual-quaternions is computed as

$$\begin{aligned}\check{q}_1\check{q}_2 &= (q_1 + \epsilon p_1)(q_2 + \epsilon p_2) \\ &= q_1q_2 + \epsilon(q_1p_2 + p_1q_2) &(2)\\ &= [\check{q}_1]_+\check{q}_2 &(3)\\ &= [\check{q}_2]_-\check{q}_1, &(4)\end{aligned}$$

where $[\check{q}]_+$ and $[\check{q}]_-$ are the left- and right-product matrices of a dual-quaternion respectively, defined as

$$\begin{aligned}[\check{q}]_+ &:= \begin{bmatrix} [q]_+ & 0_{4\times4} \\ [p]_+ & [q]_+ \end{bmatrix} \\ [\check{q}]_- &:= \begin{bmatrix} [q]_- & 0_{4\times4} \\ [p]_- & [q]_- \end{bmatrix},\end{aligned} \tag{5}$$

in which, $[q]_+$ and $[q]_-$ are the standard left- and right-product matrices of a quaternion defined as

$$[q]_+ := \left[\begin{array}{c|ccc} q_0 & -q_1 & -q_2 & -q_3 \\ \hline q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{array}\right] \tag{6}$$

$$[q]_- := \left[\begin{array}{c|ccc} q_0 & -q_1 & -q_2 & -q_3 \\ \hline q_1 & q_0 & q_3 & -q_2 \\ q_2 & -q_3 & q_0 & q_1 \\ q_3 & q_2 & -q_1 & q_0 \end{array}\right]. \tag{7}$$

Please note that the matrix representations of quaternions and dual-quaternions are for the convenience of the product operations and are thus adopted in this work.

Similar to the unit quaternion, a unit dual-quaternion satisfies the unit norm,

$$\check{q}^*\check{q} = 1 \tag{8}$$

where $\check{q}^* = q^* + \epsilon p^*$ is the conjugate dual-quaternion of $\check{q}$. Substituting to (1), we can obtain two intrinsic constraints of the unit dual-quaternion

$$\|q\| = 1, \quad q^*p = 0. \tag{9}$$

It is well known that a unit dual-quaternion has specific meanings for the rigid body transformation, which are nicely presented by the following properties

[1] $f(q + \epsilon p) = f(q) + \epsilon\dot{f}(p)$ for arbitrary quaternions q and p as $\epsilon^2 = \epsilon^3 = \cdots = 0$.

[2] Both of the algebraic and vector forms of the quaternion will be used interchangeably in this work

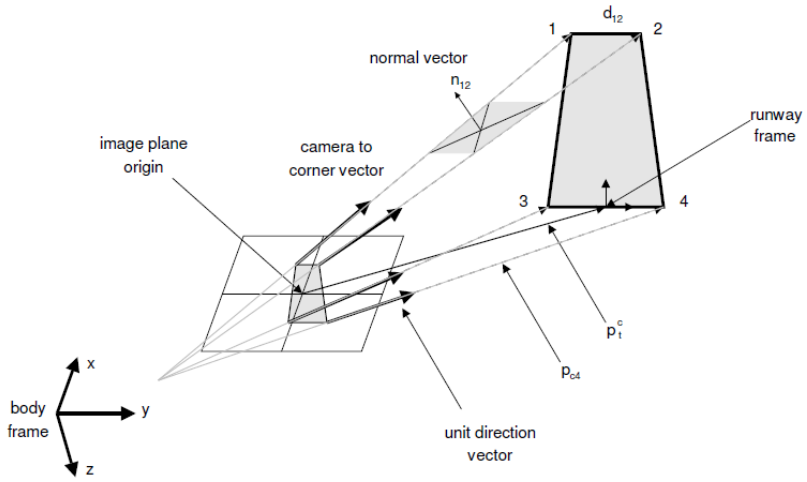

Fig. 2. The sensing geometry of visual runway observation, showing the relationship between the image and runway corner position vectors.

Property 1: Let $\mathbf{t} = [x, y, z]^T$ and q are the translation vector and quaternion of a rigid body with respect to a global frame, respectively, then the rigid-body transformation can be representated as a following unit dual-quaternion [17],

$$\check{q} = q + \epsilon p, \quad \text{with} \quad p = \frac{1}{2} t q, \tag{10}$$

where $t = [0, \mathbf{t}]^T$ is the quaternion form of the vector $\mathbf{t}$.

III. Runway Observation using Unit-dual-quaternion

The pose measurement of the runway $\mathbf{z}$ is parametrized in unit dual-quaternion form $\check{q}$,

$$\mathbf{z}(q + \epsilon p) = [x,\ y,\ z,\ q_0,\ q_1,\ q_2,\ q_3]^T, \tag{11}$$

which are obtained from the onboard vision and inertial sensors. Each of the four corner locations in the image can be transformed into unit direction vectors using the known optical properties of the sensor. This relationship is illustrated in Fig. 2. The objective is to derive the equations that relate the aircraft rotation and translation relative to the runway. The runway state relative to the navigation frame can then be obtained through its relationship with the aircraft. However, the relative rotation and translation of the aircraft relative to the runway must first be ascertained from the corner observations. This is achieved using dual-quaternions, and the output of this transformation process is used as the runway observation. The unit directional corner position vectors provide a unique solution to the aircraft-runway attitude. By assuming a pinhole-camera model, each vector is obtained from the (x, y) coordinates of its respective corner on the image plane and, the focal length of the camera (in pixels). The four unit-vectors in the camera coordinate system, denoted $\{\mathbf{p}_i^c \in \mathbb{R}^3\}_{i=\{1\ldots4\}}$, are precipitated through a normalization process.

The problem is formulated by choosing any two unit vectors, $\{\mathbf{p}_i^c, \mathbf{p}_j^c\}_{i \neq j}$, the plane defined by them, and the unit normal vector to this plane, denoted $\mathbf{n}_{ij}^c$. It is obtained by taking the cross product of $\{\mathbf{p}_i^c, \mathbf{p}_j^c\}$ and normalizing the result. Also required is the unit vector, $\mathbf{d}_{ij}^n$, that describes the direction of the vector created by subtracting one position vector from the other, relative to the runway frame. From the definitions, it can be observed that $\mathbf{d}_{ij}^n \in \text{span}\{\mathbf{p}_i^n, \mathbf{p}_j^n\}$. Then, it is necessarily the case that $\mathbf{n}_{ij}^c \cdot \mathbf{d}_{ij}^c = 0$. This provides the constraint

$$\mathbf{n}_{ij}^c \cdot \left(\mathbf{R}_n^c \mathbf{d}_{ij}^n\right) = 0, \tag{12}$$

where the direction vector has been given with respect to the runway navigation frame, and the rotation matrix transforms a vector in the navigation frame to the camera frame,

$$(\mathbf{R}_n^c)^T(q) = \begin{bmatrix} 1-2(q_2^2+q_3^2) & 2(q_1q_2+q_0q_3) & 2(q_1q_3-q_0q_2) \\ 2(q_1q_2-q_0q_3) & 1-2(q_1^2+q_3^2) & 2(q_2q_3+q_0q_1) \\ 2(q_1q_3+q_0q_2) & 2(q_2q_3-q_0q_1) & 1-2(q_1^2+q_2^2) \end{bmatrix}.$$

A rectangular runway consists of two sets of parallel edges, where each set is parallel to an axis of the runway frame. Thus, there are six possible forms for $\mathbf{d}_{ij}^n$ to take, only four of which are required to satisfy the constraints

$$\mathbf{d}_{ij}^n = \begin{cases} (1 & 0 & 0)^T \\ (0 & 1 & 0)^T \\ (\frac{1}{2}x_1^n & y_1^n & 0)^T \\ (\frac{1}{2}x_1^n & y_2^n & 0)^T \end{cases}, \tag{13}$$

where the third and fourth are obtained from the two diagonal combinations of corners $(1, 4)$ and $(2, 3)$. The second equation is obtained through a transformation and requires the dimensions of the runway to be known.

The position of the j^{th}-corner, relative to the center of the runway frame $\mathbf{t}^c$, is $\mathbf{p}_j^n$. This can be expressed relative to the camera frame as

$$\mathbf{p}_j^c = \mathbf{R}_n^c \mathbf{p}_j^n + \mathbf{t}^c, \tag{14}$$

which must also be perpendicular to $\mathbf{n}_{ij}^c$, producing a second constrain

$$\mathbf{n}_{ij}^c \cdot (\mathbf{R}_n^c \mathbf{p}_j^n + \mathbf{t}^c) = 0. \tag{15}$$

Expanding $\mathbf{n}_{ij}^c$ and substituting into Eqs. 12 and 15 produces the two complete homogeneous equations

$$\left(\frac{\mathbf{p}_i^c \times \mathbf{p}_j^c}{||\mathbf{p}_i^c \times \mathbf{p}_j^c||}\right) \cdot \mathbf{R}_n^c \mathbf{d}_{ij}^n = 0 \tag{16}$$

$$\left(\frac{\mathbf{p}_i^c \times \mathbf{p}_j^c}{||\mathbf{p}_i^c \times \mathbf{p}_j^c||}\right) \cdot (\mathbf{R}_n^c \mathbf{p}_j^n + \mathbf{t}^c) = 0. \tag{17}$$

The solving of which produces the position $\mathbf{t}^c$ and attitude $\mathbf{R}_n^c$ of the camera relative to the runway frame.

In this application, the solutions are obtained by reformulating Eqs. 16 and 17 using a unit-dual quaternion, $\check{q} = q + \epsilon p$, as the error functions of the Levenberg-Marquardt optimization. The reformulation process necessitates the transformation of unit vectors and rotation matrices into their quaternion equivalents, in which a vector $\mathbf{v}$ changes to a quaternion equivalent form $v = [0, \mathbf{v}^T]^T$, and the rotational operation of a vector $\mathbf{v}^n$ to $\mathbf{v}^c$ becomes $v^c = q^* v^n q$. By utilizing the matrix form of the quaternions, the alternative representations of Eqs. 16 and 17 become,

$$\begin{aligned} \mathbf{n}_{ij}^c \cdot \left(\mathbf{R}_n^c \mathbf{d}_{ij}^n\right) &= (n_{ij}^c)^T \left([q]_+^T [d_{ij}^n]_+ q\right) \\ &= q^T \left([n_{ij}^c]_-^T [d_{ij}^n]_+\right) q \end{aligned} \tag{18}$$

$$\begin{aligned} \mathbf{n}_{ij}^c \cdot (\mathbf{R}_n^c \mathbf{p}_j^n + \mathbf{t}_r) &= q^T \left([n_{ij}^c]_-^T [d_{ij}^n]_+\right) q \\ &\quad + q^T \left(2[n_{ij}^c]_+^T\right) p, \end{aligned} \tag{19}$$

where p is the dual part of the dual-quaternion $\check{q}$, given by $p = \frac{1}{2} t q$.

The inclusion of the unit-magnitude constraints of the dual-quaternion $q^T q = 1$ and $q^T p = 0$ (in a vector form) yields the cost functional,

$$J(\check{q}) = \sum_{i=1}^{6} \begin{pmatrix} \left(q^T [n_{ij}^c]_-^T [d_{ij}^n]_+ q\right)^2 \\ + \left(q^T [n_{ij}^c]_-^T [d_{ij}^n] q + q^T 2[n_{ij}^c]_+^T p\right)^2 \\ + \lambda_1 \left(q^T q - 1\right)^2 + \lambda_2 \left(q^T p\right)^2 \end{pmatrix}, \tag{20}$$

where λ_1 and λ_2 are Lagrangian multipliers, and the objective is to select values of q and p that minimizes the error output, which in this case, is zero. The key property of this cost functional is that the error function is separable, allowing q to be solved independently of p. Its alternate form is given by

$$J_1(q) = \sum_{i=1}^{6} \left(q^T [n_{ij}^c]_-^T [d_{ij}^n]_+ q\right)^2 + \lambda_1 \left(q^T q - 1\right)^2 \tag{21}$$

$$J_2(q,p) = \sum_{i=1}^{6} \begin{pmatrix} \left(q^T [n_{ij}^c]_-^T [d_{ij}^n] q + q^T 2[n_{ij}^c]_+^T p\right)^2 \\ + \lambda_2 \left(q^T p\right)^2 \end{pmatrix}, \tag{22}$$

where the real-valued quaternion q is the first to be obtained through Levenberg-Marquardt algorithm. Using the Jacobian $\nabla J_1(q)$, the Levenberg-Marquardt algorithm searches in the direction given by the estimated solution $\hat{q}(k)$ to the equation,

$$[\nabla J_1^T \nabla J_1 + \lambda I] \delta\hat{q}(k) = -(\nabla J_1^T) J_1(\hat{q}(k)) \tag{23}$$

where $\delta q(k)$ is the estimated quaternion error at k-iteration, and λ is a nonnegative scalar and I is the identity matrix, while the Jacobian matrix $\nabla J_1(\hat{q}(k))$ evaluates to

$$\nabla J_1(\hat{q}(k)) = \sum_{i=1}^{6} \begin{pmatrix} 2\left(\hat{q}^T [n_{ij}^c]_-^T [d_{ij}^n]_+ \hat{q}\right) \times \\ \left(\hat{q}^T [n_{ij}^c]_-^T [d_{ij}^n]_+ + \hat{q}^T [n_{ij}^c]_-^T [d_{ij}^n]_+^T\right) \\ + 4\lambda_1 \left(\hat{q}^T \hat{q} - 1\right) \hat{q}^T \end{pmatrix}. \tag{24}$$

During the operation, the estimate $\delta\hat{q}(k)$ is used to update $\hat{q}(k)$, and the estimate reduces with successive iterations. Convergence is achieved when the magnitude of the cost functional is less than a preset threshold (10^{-4} is used in this work). For the generic case, the LM algorithm is guaranteed to converge but may not necessarily converge to the global minimum of the function. In this particular application, however, they are one and the same.

The estimate of the dual part of the quaternion $p(k)$ can be obtained by substituting the final attitude estimate $q(k)$ into the Jacobian of Eq. 20 or Eq. 21. Alternatively, it can be obtained through its relationship with $q(k)$

$$p(k) = -(A_1 + A_2^T)^{-1} A_2 q(k), \tag{25}$$

with

$$A_1 = \left(\sum_i^k \beta_i\right) I \tag{26}$$

$$A_2 = 2 \sum_i^k \beta_i ([p_i^n]_+ - [p_i^n]_-) q(k). \tag{27}$$

The unit dual-quaternion $\check{q} = q + \epsilon p$ of the runway is thus obtained as the output of the LM optimization. If necessary, $\mathbf{t} = [x, y, z]^T$ can be computed from the unit dual-quaternion by utilizing Eq. 10.

IV. Runway-Aided Inertial Navigation

To demonstrate the effectiveness of the runway pose observation, a runway-aided inertial navigation system is designed. The nominal inertial navigation model is a simplified one utilizing a local-fixed, local-tangent navigation frame which is suitable for low-quality inertial sensor applications. An integration Extended Kalman filter consists of an inertial state vector $\mathbf{x} = [(\mathbf{t}^n)^T, (\mathbf{v}^n)^T, (q^n)^T]^T$ with a state kinematic model,

$$\begin{aligned} \dot{\mathbf{t}}^n &= \mathbf{v}^n + \mathbf{w}_1 \\ \dot{\mathbf{v}}^n &= \mathbf{R}_b^n \mathbf{f}^b + \mathbf{g}^n + \mathbf{w}_2 \\ \dot{q}^n &= \frac{1}{2} q^n \omega^b, \end{aligned}$$

where

- Position in the navigation frame $\mathbf{t}^n = [x, y, z]^T$
- Vehicle in the navigation frame $\mathbf{v}^n = [v_x, v_y, v_z]^T$
- Accelerometer measurement $\mathbf{f}^b = [f_x, f_y, f_z]^T$
- $\mathbf{g}^n$ is the gravitational acceleration
- $\mathbf{R}_b^n$ is a transforming matrix of a vector from body to navigation frame
- Attitude quaternion $q^n = [q_0, q_1, q_2, q_3]^T$

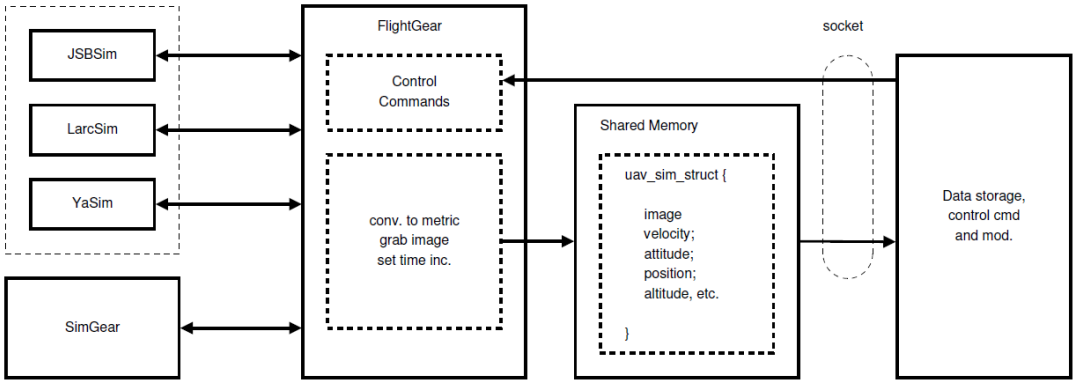

Fig. 3. Flowchart of the internal modi cations to FlightGear, data structure and socket interface.

- Gyroscope measurement $\omega^b = [0, \omega_x, \omega_y, \omega_z]^T$ (in a quaternion form)
- $(\mathbf{w}_1, \mathbf{w}_2)$ are the white Gaussian processes noises with strength matrices of $\boldsymbol{Q}_1, \boldsymbol{Q}_2$ for the translation and velocity, respectively,

The runway pose solutions $\mathbf{z}$ from the previous section are subsequently used as the observations to the extended Kalman filter, which are related to the inertial navigation state

$$\mathbf{z} = \begin{bmatrix} \mathbf{t} \\ q \end{bmatrix} = \begin{bmatrix} \mathbf{R}_b \mathbf{R}_n^b \mathbf{t}^n + \mathbf{v}_1 \\ q(\mathbf{R}_b \mathbf{R}_n^b) + \mathbf{v}_2 \end{bmatrix},$$

with $(\mathbf{v}_1, \mathbf{v}_2)$ are the white Gaussian observation noises with strength matrices of $\boldsymbol{R}_1, \boldsymbol{R}_2$ for the translation and quaternion, respectively. By using the nonlinear process and observation models and the corresponding Jacobian matrices, the state vector and its covariance are propagated and updated.

V. Experiment Results

A. FlightGear Interface

The FlightGear package is a continuously developing open-source flight simulator[3]. The complete package contains the simulation engines, called JSBSim, Yasim and LarcSim, and SimGear (math simulation, navigation and data management package). Both JSBSim and SimGear were initiated with FlightGear and are developed in concert with it. The FlightGear program itself serves as the front-end to the simulator. Its primary purpose is to interface between the user and the underlying software, relaying control commands and interpreting simulator outputs for controlling the 3D graphical user interface (GUI). The co-development of FlightGear, SimGear, and JSBSim made their software amenable to modification. To be suitable for an experimental testbed, the software was modified as outlined in Fig. 3. The first modification was the extraction of aircraft data and images. The required data set was stored in a shared memory location accessible through a socket. The second was an internal time modification to ensure FlightGear ran in a real-time manner. Depending on the additional processes used on the image obtained from FlightGear, such as feature extraction, control algorithms, and so on, the image rate may slow considerably. When an image and data set has been retrieved, FlightGear is temporarily put on hold until control is given back to it through the socket. Thus, the Δt parameter passed to the simulator engine from FlightGear must be modified to account for the time FlightGear is paused. By setting this parameter to the reciprocal of the frame rate, the simulation ensures that the aircraft flies in real-time approximately. Thus, the time required for external processing does not affect the simulation. However, the amount of computation requires careful monitoring as the control algorithms will function erratically as Δt is increased. The storage of data and images and their subsequent processing necessitated the development of numerous algorithms. Several of these were used in conjunction with FlightGear, with many used for the post-processing of data after the completion of the experiments. During runtime, rendered images were extracted from FlightGear and saved in an image format, while the instantaneous state information was stored as one large text. The images were also displayed in an open-source GUI display tool called EZWGL. This allowed the functionality of algorithms to be monitored during runtime. A typical output is illustrated in Fig. 1. Two images are shown in the figure. On the left is the raw grey-scale image extracted from FlightGear and on the right, is the same image with its corners highlighted. Clearly shown are the four runway corners and those due to the white runway markings. Although only the corners marking the extremities of the runway were used in this application, the addition of all corners on the runway would improve the state estimate.

The experimental procedure was initialized with the startup of the modified FlightGear software package and the data acquisition software. Once communications had been established through the socket, the simulation commenced. During runtime, the images extracted from FlightGear were displayed in the EZWGL interface, along with their corner detected counterparts. In an attempt to make the simulation more realistic and to extend the duration of each experiment, an image was recorded for every tenth data set. This had the effect of reducing the time taken to store images, the

[3]http://www.flightgear.org/

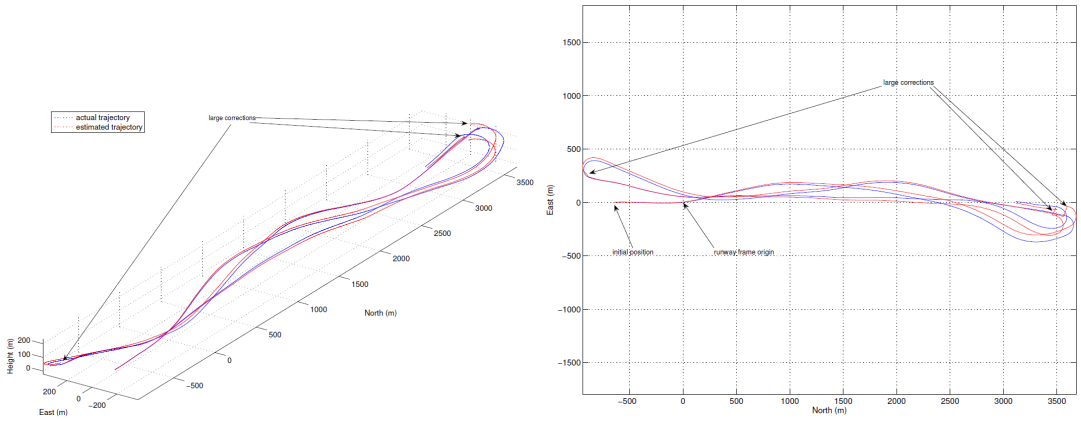

Fig. 4. (left) The actual 3D trajectory traversed and (right) 2D trajectory traversed, showing the regions where the four runway corners were in view.

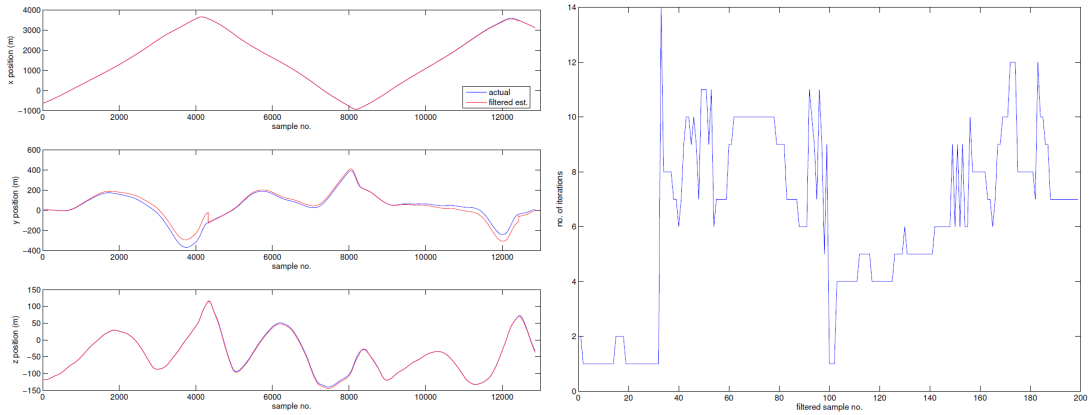

Fig. 5. (left) The estimated and actual position and (right) the number of iterations required by the Levenberg-Marquardt minimization algorithm.

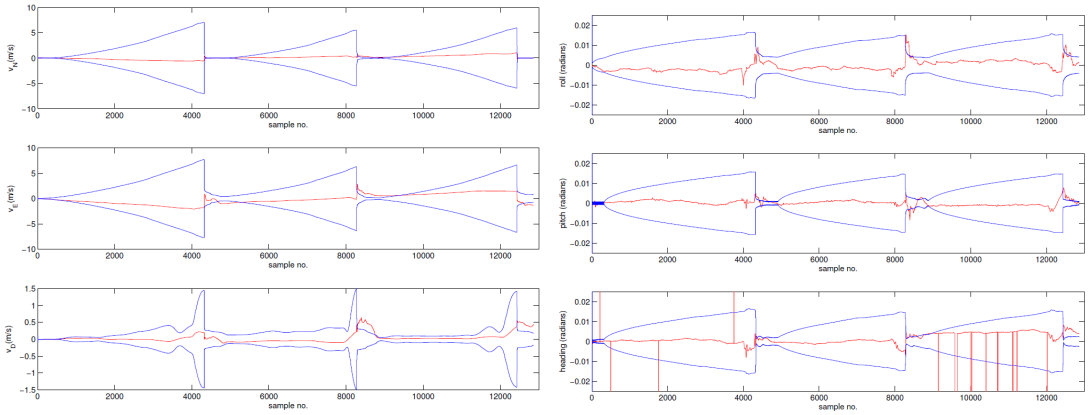

Fig. 6. The filter's error state in velocity and attitude with 1σ uncertainty (the position error state looks similar to that of velocity and thus is omitted.

memory required to do so and, more closely approximated the data to image rate that would be acquired in a UAV implementation. During the experiments, the aircraft was controlled by the user in making several lengthwise passes over the runway, ensuring that the complete runway appeared in the image for as long as possible. At the completion of each experiment, the image and data sets were inspected for completeness, compressed and stored. During runtime, the corner extraction was performed using the SUSAN [39] corner detecting algorithm, chosen for its high performance, adaptability and robustness to noise.

Initial processing acquired the identity of the images that contained the four corners of the runway. State estimation was performed using initial conditions obtained from the simulation, with noise added to necessary body derived measurement vectors. When the identity of the input vector happened to coincide with an image, the identity of the image was compared with the list of images known to contain the

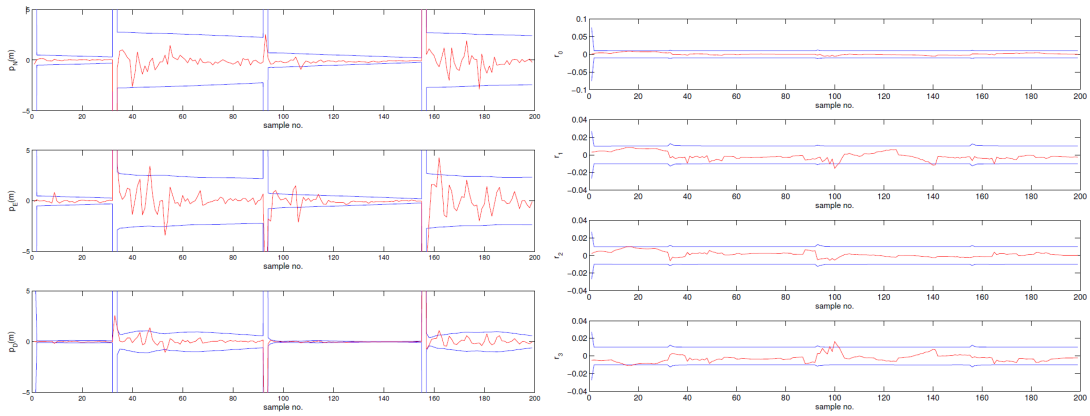

Fig. 7. The filter's position and attitude innovation with 1σ uncertainty.

runway. If the complete runway was contained in the image, then the set of camera-to-corner unit vectors was ascertained. Interestingly, this is one instance where the addition of noise was not mandatory. This is because the simulator must first discretize the position and dimensions of the runway before rendering them in an image with a finite number of pixels. This process introduces quantization errors into each of the corner positions which cascades through the filtering process. Subsequent to the acquisition of the direction vectors, a cost function was formulated using dual-quaternions and the known dimensions of the runway. A Levenberg-Marquardt minimization algorithm was produced from the cost function and separated to provide a solution to the rotation.

B. Results

Fig. 4 compares the estimated and actual 3D and 2D paths traversed by the aircraft showing the correctional effects when the runway was in full view of the camera. Fig. 5(left) further compares the estimated aircraft xyz-position with the actual obtained during the experiment. The effect of adding Gaussian noise to the accelerometer and gyroscope data is most noticeable in the position plot. This is clearly evident in the aircraft's y position as the y position appears to be affected the most because of the low magnitudes relative to the other two axes. Fig. 5(right) illustrates the number of iterations required for convergence of the Levenberg-Marquardt process through the usable range of images. Convergence was assumed when the residual was less than the value of 10^{-2}. Thresholds lower than this are not guaranteed to produce better results. A weighting of 100 was placed on the $(q^T q - 1)$ term in the cost function for the faster convergence. Fig. 6 shows the estimated errors in velocity and attitude (converted to Euler angles for display) with 1σ uncertainty. Fig. 7 provides the filter's innovation sequences of the position and attitude with 1σ uncertainty, showing the consistent operations of the filter.

VI. Conclusions

This work presented the development of an algorithm that allows the use of visual runway observations as zero bias, absolute navigation references. Runways are suitable for this task because their geodetic coordinates are known precisely, as are their dimensions and detection markings, and the fact that they are designed to be highly visible. The emphasis in this application was placed on increasing the level of situational awareness for an aircraft during its landing approach, providing additional redundancy to navigating in the vicinity of a runway, thereby reducing the reliance on GPS. The algorithm produces a unit dual-quaternion based cost function based on the geometric relationships between the observations and the runway, and an iterative minimization of the cost function using the Levenberg-Marquardt technique. The algorithm was then integrated into the inertial navigation system and demonstrated using a flight simulator using a camera-in-the-loop setup. Future work is extending the method to accommodate the state-of-the-art nonlinear observer-based SLAM technique and the unit-dual quaternion to enhance stability and efficiency.

References

[1] J. Kim, J. Guivant, M. L. Sollie, T. H. Bryne, and T. A. Johansen, "Compressed pseudo-slam: Pseudoranges-integrated compressed simultaneous localisation and mapping for uav navigation," *THE JOURNAL OF NAVIGATION*, vol. 1, 2020 (in press).

[2] V. Usenko, N. Demmel, D. Schubert, J. Stckler, and D. Cremers, "Visual-inertial mapping with non-linear factor recovery," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 422–429, April 2020.

[3] Q. Liang and M. Liu, "A tightly coupled vlc-inertial localization system by ekf," *IEEE Robotics and Automation Letters*, pp. 1–1, 2020.

[4] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, Feb 2017.

[5] M. G. Miiller, F. Steidle, M. J. Schuster, P. Lutz, M. Maier, S. Stoneman, T. Tomic, and W. Strzl, "Robust visual-inertial state estimation with multiple odometries and efficient mapping on an mav with ultra-wide fov stereo vision," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018, pp. 3701–3708.

[6] R. Ashford, "Flight safety digest: A study of fatal approach-and-landing accidents worldwide 1980-1996," *Flight Safety Digest*, vol. 17, no. 2/3, Feb-March 1998.

[7] P. Serra, R. Cunha, T. Hamel, D. Cabecinhas, and C. Silvestre, "Landing of a quadrotor on a moving target using dynamic image-based visual servo control," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1524–1535, Dec 2016.

[8] L. Xu and H. Luo, "Towards autonomous tracking and landing on moving target," in *2016 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, June 2016, pp. 620–628.

[9] Daquan Tang, Yongkang Jiao, and Jie Chen, "On automatic landing system for carrier plane based on integration of ins, gps and vision," in *2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC)*, Aug 2016, pp. 2260–2264.

[10] K. D. Nguyen and C. Ha, "Vision-based hardware-in-the-loop-simulation for unmanned aerial vehicles," in *Intelligent Computing Theories and Application*. Springer International Publishing, 2018, pp. 72–83.

[11] L. Coutard, F. Chaumette, and J.-M. Pflimlin, "Automatic landing on aircraft carrier by visual servoing," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 2843–2848.

[12] M. Kus *et al.*, "Autonomous carrier landing of a fixed-wing uav with airborne deck motion estimation," Ph.D. dissertation, 2019.

[13] V. Gibert, F. Plestan, L. Burlion, J. Boada-Bauxell, and A. Chriette, "Visual estimation of deviations for the civil aircraft landing," *Control Engineering Practice*, vol. 75, pp. 17–25, 2018.

[14] A. J. Moore, M. Schubert, C. Dolph, and G. Woodell, "Machine vision identification of airport runways with visible and infrared videos," *Journal of Aerospace Information Systems*, vol. 13, pp. 266–277, 2016.

[15] Y. Wu, X. Hu, D. Hu, T. Li, and J. Lian, "Strapdown inertial navigation system algorithms based on dual quaternions," *IEEE transactions on aerospace and electronic systems*, vol. 41, no. 1, pp. 110–132, 2005.

[16] X. Wang, C. Yu, and Z. Lin, "A dual quaternion solution to attitude and position control for rigid-body coordination," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1162–1170, Oct 2012.

[17] J. Cheng, J. Kim, J. Shao, and W. Zhang, "Robust linear pose graph-based SLAM," *Robotics and Autonomous Systems*, vol. 72, pp. 71–82, 2015.

[18] Ping Han, Zheng Cheng, and Ling Chang, "Automatic runway detection based on unsupervised classification in polsar image," in *2016 Integrated Communications Navigation and Surveillance (ICNS)*, April 2016, pp. 6E3–1–6E3–8.

[19] K. Abu-Jbara, W. Alheadary, G. Sundaramorthi, and C. Claudel, "A robust vision-based runway detection and tracking algorithm for automatic uav landing," in *2015 International conference on unmanned aircraft systems (ICUAS)*. IEEE, 2015, pp. 1148–1157.